Evaluating the Accuracy and Efficiency of Sentiment Analysis Pipelines with UIMA*

Nabeela Altrabsheh, Georgios Kontonatsios, and Yannis Korkontzelos

Department of Computer Science, Edge Hill University, Ormskirk, United Kingdom
{altrabsn, Georgios.Kontonatsios, Yannis.Korkontzelos}@edgehill.ac.uk

Abstract. Sentiment analysis methods co-ordinate text mining components, such as sentence splitters, tokenisers and classifiers, into pipelined applications to automatically analyse the emotions or sentiment expressed in textual content. However, the performance of sentiment analysis pipelines is known to be substantially affected by the constituent components. In this paper, we leverage the Unstructured Information Management Architecture (UIMA) to seamlessly co-ordinate components into sentiment analysis pipelines. We then evaluate a wide range of different combinations of text mining components to identify optimal settings. More specifically, we evaluate different pre-processing components, e.g. tokenisers and stemmers, feature weighting schemes, e.g. TF and TFIDF, feature types, e.g. bigrams, trigrams and bigrams+trigrams, and classification algorithms, e.g. Support Vector Machines, Random Forest and Naive Bayes, against 6 publicly available datasets. The results demonstrate that optimal configurations are consistent across the 6 datasets while our UIMA-based pipeline yields a robust performance when compared to baseline methods.

Keywords: sentiment analysis, text processing, interoperability, UIMA

1 Introduction

The Unstructured Information Management Architecture (UIMA) [4] is a software framework that facilitates the development of interoperable text mining applications. UIMA-enabled components can be freely combined into larger pipelined applications, e.g. machine translation [8] and information extraction, using UIMA's common communication mechanism and shared data type hierarchy, i.e. Type System. Recent studies has demonstrated that UIMA-based pipelines can efficiently address a wide range of different text mining tasks [2, 8].

In this paper, we use the UIMA framework to develop efficient sentiment analysis pipelines. We focus on sentiment analysis, considering that automatic sentiment analysis systems are being increasingly used in a number of applications, such as business and government intelligence. The popularity of the task can largely be associated with the vast amount of available data, especially in social media. For example, sentiment analysis on Twitter has been used to identify concerns in urban enviroments [19].

* The final authenticated version is available online at https://doi.org/10.1007/978-3-030-23281-8_23.

Despite the popularity of sentiment analysis and the wide applicability of UIMA to many text processing tasks, UIMA has been used for sentiment analysis by a few studies, only. Rodriguez et al. [13] developed UIMA-based pipelines for capturing the sentiment expressed in customers' reviews about hotels.

This study investigates sentiment analysis using the UIMA framework. Further than Rodriguez et al. [13], (a) we investigate the effect of different pre-processing components, features, and feature selection on the overall performance of a sentiment analysis system, and (b) we compliment evaluation results with the execution times of each combination of components and classifiers. Our results show that execution times vary widely and that high execution times do not always match high accuracies. To the best of our knowledge, this is the first work that considers execution times while evaluating UIMA pipelines. The execution time of a sentiment analysis system is particularly important for real-time applications, especially when monitoring social media.

2 Related Work

The Unstructured Information Management Architecture (UIMA) has been employed widely for developing text processing applications in various domains. Kontonatsios et al. [8] extended UIMA workflows to facilitate the creation of multilingual and multimodal NLP applications. In the medical domain, UIMA has been applied to detect the smoking status of patients [17]. UIMA has been used to analyse hotel customer reviews [13], where sentiment analysis is modelled as a classification task. UIMA was shown to be suitable for designing and implementing sentiment analysis systems due to the reusability components.

Several studies have explored the time that classifiers take to identify polarity. For instance, Greaves et al. [6], who researched sentiment analysis to analyse patients' experience, concluded that the Naive Bayes Multinomial classifier was faster than other classifiers by a short margin of 0.2 seconds. Of course, data size can affect the model's running time. Running large datasets using limited computational resources can cause out-of-memory errors, and distributing the training task across many machines was shown to decrease running time by 47% [7]. Apart from classifier training, other components the pipeline, parameters and feature types can also affect execution times [5].

3 Experiments

As any other UIMA application, our sentiment analysis pipeline implements three basic operations: read (*Collection Reader*), process (*Analysis Engine*) and write (*CAS Consumer*). We have conducted 6 large-scale experiments to investigate the optimal pipeline configuration. More specifically, we evaluated all combinations of the following components: 1) CoreNLP and Snowball Tartarus stemmers, 2) TF and TF-IDF feature weighting schemes, 3) feature types: unigrams, bigrams, trigrams and combinations of them, 4) frequency thresholds for feature filtering, i.e. feature removal, and 5) classification algorithms: Support Vector Machines, Random Forest and Naive Bayes, as implemented in the WEKA platform. It should be noted that different pipeline configurations were created by simply changing the UIMA XML descriptor file.

Table 1. Data Sources

	Amazon [9]	**IMDB [9]**	**SemEval [14]**	**Senti-140 [10]**	**UMICH [18]**	**Yelp [9]**
Dataset Type	Training (subset)	Training (subset)	Development set	Training (subset)	Training set	Training (subset)
size	1,000	1,000	20,632	1,048,575	7,086	1,000
positive	500	500	7059	554,470	3,995	500
negative	500	500	3,231	494,105	3,091	500
neutral	-	-	10,342	-	-	-

* **SemEval** refers to Task 4A of SemEval 2016.

All combinations of the components above are evaluated in terms of accuracy (*Acc*), precision (*P*), recall (*R*) and F-score (*F1*) using 10-fold cross validation. In addition, we measured the execution time of each pipeline configuration. We used 6 publicly available datasets. Table 1 shows the source, name, size and number of documents labelled as positive, negative or neutral in each dataset. The neutral label is only available in the *SemEval* dataset and we did not include it in our experiments. Amazon, IMDB, UMICH and Yelp experiments were run on a HP laptop with Intel core i5-8250u, 1.80GHz, on Windows. SemEval and Senti-140 experiments were run on an HP ProLiant DL360 Gen9 server running Linux.

The first experiment evaluates our sentiment analysis pipeline when using different combinations of pre-processing components. We use UIMA to plug and play preprocessing components into pipelines, while using the same type-system, to identify the best configuration. Many studies explored the effect of preprocessing on sentiment analysis. Preprocessing can improve performance up to 20%, while analysing sentiment in students' feedback [1]. We develop 4 pipelines by combining 2 tokenisers and 2 stemmers, common in the literature: 1) Standard tokeniser (T1): segments a document into its tokens using whitespace characters as delimiter. This tokeniser was implemented in-house, 2) StringTokenizer (T2): from the java.util package[1], 3) englishStemmer (S1): from the tartarus.snowball package[2], and 4) PorterStemmer (S2): from the tartarus.snowball package[3]. The first experiment evaluates 120 configurations: 2 tokenisers x 2 stemmers x 1 ngrams (unigrams+bigrams+trigrams combined) x 6 datasets x 5 classifiers. The remaining experiments use the best performing combination.

The second experiment considers two feature weighting schemes: Term Frequency (TF) and Term Frequency Inverse Document Frequency (TF-IDF). TF and TF-IDF are different ways of assessing feature importance by assigning different weights.

Choosing features that represent data instances accurately for a particular task can lead to more accurate predictions. The most common feature types used for sentiment analysis are n-grams, i.e. sequences of n textual units, which can be letters, syllables or words [1]. N-grams usually consider tokens and are of one, two or three tokens long, i.e. unigrams, bigrams or trigram, respectively. Sarker et al. [15] and Pal and Gosh [11] used n-gram features for developing sentiment analysis methods and evaluated their methods against the same datasets that we use in this work. Here, we explore the fol-

[1] docs.oracle.com/javase/7/docs/api/java/util/StringTokenizer.html
[2] snowball.tartarus.org/algorithms/english/stemmer.html
[3] snowball.tartarus.org/algorithms/porter/stemmer.html

Table 2. Best pipeline configurations in terms of both F-Score and execution time across the 6 evaluation datasets. The table also reports the highest and lowest F-Score and the slowest and fastest execution time obtained by the different pipeline configurations.

		Pipeline Configuration	F1	Time (mm:ss)		Pipeline Configuration	F1	Time (mm:ss)
Highest F-Score	Amazon	CNB-T1-S1	.831	00:01	IMDB	CNB-T1-S1	.787	00:02
Lowest F-Score		NB-T1-S1	.748	00:05		NB-T2-S1	.675	01:00
Slowest Time		RF-T1-S2	.777	06:09		RF-T2-S1	.699	05:13
Fastest Time		CNB-T2-S2	.829	00:01		CNB-T1-S2	.773	00:02
Best Configuration		**CNB-T1-S1**	**.831**	**00:01**		**CNB-T1-S1**	**.787**	**00:02**
Highest F-Score	SemEval	CNB-T1-S1	.832	00:02	Senti-140	LIB-T1-S1	.798	02:03:42
Lowest F-Score		NB-T2-S2	.588	07:06		CNB-T2-S1	.768	27:04
Slowest Time		RF-T1-S1	.753	01:22:05		LIB-T2-S2	.796	02:07:00
Fastest Time		CNB-T1-S1	.808	00:01		CNB-T1-S2	.779	26:20
Best Configuration		**CNB-T1-S1**	**.832**	**00:02**		**CNB-T1-S2**	**.779**	**00:25**
Highest F-score	UMICH	RF-T2-S1	.998	17:46	Yelp	CNB-T1-S2	.798	00:01
Lowest F-Score		NB-T2-S1	.807	04:11		NB-T2-S1	.665	01:02
Slowest Time		RF-T1-S2	.997	22:23		RF-T2-S2	.745	04:35
Fastest Time		CNB-T2-S2	.979	00:01		RF-T2-S2	.745	04:35
Best Configuration		**SVM-T2-S1**	**.991**	**00:05**		**CNB-T1-S2**	**.798**	**00:01**

lowing n-gram combinations: unigrams only, bigrams only, trigrams only, unigrams and bigrams, unigrams and trigrams, bigrams and trigrams, and all n-grams combined.

The fourth experiment evaluates our pipeline when filtering features using a frequency threshold. Considering a research objective is to scale text processing pipelines to big data collections, we are interested in reducing the computational resources needed to execute them without reducing the accuracy of the underlying text mining models. Equal thresholds were set for all ngram features, and we experimented with threshold values in the range of $[1, 30]$. We aim to remove infrequent features to eliminate potential noise in the datasets. Running times are expected to decrease as threshold values increase. If the performance of the models does not decrease significantly as threshold values increase, then high values can safely be adopted, leading in models of smaller size that are easier to transfer and work with, without loss in prediction accuracy.

The choice of a classifier substantially affects the performance of the sentiment analysis pipeline. We experiment with the following classifiers: SVM, NB, RF, CNB and LibLinear. CNB and LibLinear have not been previously evaluated on these datasets[4].

4 Results and Discussion

Table 2 shows the lowest and highest F-score and the slowest and fastest execution time achieved by the pipeline configurations. We further report the best configuration considering both the F-score performance and the execution time. As an example, we observe that SVM-T2-S1 achieves an F-score of 0.991 on the $UMICH$ dataset, which

[4] Only CNB and LIB were evaluated on Senti-140, as the other classifiers failed to run due to out-of-memory errors.

Table 3. Average performance of our sentiment analysis pipeline, on combinations of pre-processing components. The results are averaged over 5 classifiers, as discussed in section 3.

		Pipeline Configuration					Pipeline Configuration			
Metric		**T1- S1**	**T1-S2**	**T2-S1**	**T2-S2**		**T1- S1**	**T1-S2**	**T2-S1**	**T2-S2**
Accuracy	**Amazon**	.802	**.803**	.802	**.803**	**IMDB**	**.736**	.734	.719	.717
Precision		.807	.807	.807	.807		**.740**	.738	.725	.723
Recall		.802	**.803**	.802	**.803**		**.740**	.733	.719	.717
F-score		.801	**.802**	.801	**.802**		**.734**	.732	.717	.715
Accuracy	**SemEval**	**.791**	.789	.774	.777	**Senti-140**	**.786**	.785	.783	.783
Precision		**.789**	.785	.774	.779		**.789**	**.789**	.787	.787
Recall		**.760**	.758	.746	.745		**.788**	.787	.785	.784
F-score		**.757**	.756	.740	.741		**.785**	**.785**	.782	.782
Accuracy	**UMICH**	**.964**	.959	.954	.955	**Yelp**	**.765**	.764	.745	.750
Precision		**.968**	.967	.962	.962		**.770**	.769	.750	.755
Recall		.967	.958	.957	**.969**		**.770**	.764	.745	.750
F-score		**.964**	.958	.953	.955		**.770**	.764	.744	.749

is only marginally lower than the overall highest F-score, 0.998, achieved by RF-T2-S1. However, SVM-T2-S1 is our preferred configuration because it is substantially faster than RF-T2-S1. Overall, the CNB classifier obtained both a high F-score performance and a fast execution time in 5 out of 6 datasets.

Preprocessing: We evaluate 4 combinations of pre-processing components. Table 3 shows the average performance of the 4 pipeline configurations when applied to the 6 datasets. The performance is computed in terms of accuracy, precision, recall and F-score, while the reported results are average values across the performance obtained by the 5 classifiers. It can be observed that the T1-S1 configuration performed best in most cases. The improvement over the remaining configurations are insignificant.

TF & TF-IDF: TF weighting achieved slightly higher classification performance than TF-IDF in 4 out of 6 datasets, as shown in table 4. TF-IDF was faster than TF in 5 out of 6 datasets. A larger time margin, 9 seconds, was observed on the Senti-140 dataset.

Features: Table 5 shows the performance of n-gram feature combinations, introduced in section 3. The performance is computed for the best configuration (T1 and S1). Trigram features yielded the lowest performance in most cases, while the combination of all n-grams performed best in 3 out of the 6 datasets. Unigrams and trigrams together obtained the highest performance on Yelp. The performance margin between the different feature types is substantial in several occasions. For example, unigrams achieved

Table 4. Scores and execution times of CNB-T1-S1 using TF and TF-IDF feature weighting.

	Amazon		**IMDB**		**SemEval**		**Senti-140**		**UMICH**		**Yelp**	
	TF	**TF-IDF**	**TF**	**TF-IDF**	**TF**	**TF-IDF**	**TF**	**TF-IDF**	**TF**	**TF-IDF**	**TF**	**TF-IDF**
Time (sec)	.022	**.018**	.067	**.064**	1.119	**.278**	34.277	**25.061**	**.055**	.577	.387	**.020**
Acc	**.835**	.833	**.782**	.774	**.839**	.810	.772	.772	**.982**	.974	.787	**.788**
P	**.839**	.835	**.791**	.778	**.801**	.780	.780	.780	**.982**	.973	**.791**	.790
R	**.835**	.833	**.782**	.775	**.815**	.806	.777	.777	**.981**	.975	**.787**	.788
F1	**.834**	.833	**.780**	.774	**.807**	.789	.772	.772	**.982**	.974	.786	**.788**

Table 5. Features: Performance of the best configuration (T1, S1) on all datasets, preprocessing techniques and classifiers for each of the features. U: Unigrams, B: Bigrams, T:Trigrams

Metric		U	B	T	U+B	U+T	B+T	All		U	B	T	U+B	U+T	B+T	All
Accuracy	**Amazon**	.816	.704	.608	.702	**.835**	.831	.831	**IMDB**	**.816**	.635	.575	.648	.718	.807	.788
Precision		.819	.727	.676	.728	**.837**	.833	.833		**.816**	.637	.607	.650	.786	.810	.792
Recall		.816	.704	.608	.702	**.835**	.831	.831		**.816**	.635	.575	.648	.781	.807	.788
F-score		.816	.696	.567	.693	**.835**	.831	.831		**.816**	.634	.540	.647	.780	.807	.787
Accuracy	**SemEval**	.837	.773	.512	.681	.840	.821	**.844**	**Senti-140**	.738	.732	.696	.749	.768	**.877**	.772
Precision		.827	.746	.664	.707	.819	.793	**.832**		.744	.740	.714	.750	.774	.774	**.780**
Recall		.782	.775	.638	.736	.802	**.815**	.793		.743	.737	.703	.745	.772	.771	**.777**
F-score		.798	.753	.518	.676	**.809**	.801	.808		.739	.732	.694	.740	.768	.766	**.772**
Accuracy	**UMICH**	.797	.680	.691	.681	.796	**.800**	.794	**Yelp**	.930	.952	.968	.969	.975	.978	**.980**
Precision		**.819**	.688	.663	.691	.798	.801	.796		.972	.950	.966	.967	.973	.977	**.980**
Recall		.797	.680	.601	.681	.796	**.800**	.794		.973	.955	.970	.971	.975	.979	**.979**
F-score		.797	.677	.560	.677	.796	**.800**	.794		.972	.952	.968	.969	.974	.978	**.980**

an improved F-score of 27.6% over trigrams on the IMDB dataset. This suggests that careful feature selection can improve the performance of sentiment analysis pipelines.

Feature Selection: We filtered out features, i.e. n-grams, that occur less frequently than a pre-defined threshold. The results of applying threshold values in $[1, 30]$, in figure 1, show that for smaller datasets, the performance decreases as we increase the threshold. For example, the F-score on Amazon, which consists of $1,000$ reviews only, drops from 0.832 for a threshold of 1 to 0.676 for a threshold of 30. However, for larger datasets, e.g. Senti-140 that contains more than 1M documents, F-scores vary insignificantly.

Classifiers: CNB was the fastest and best. RF was the slowest, but performed best on UMICH. SVM and LIB performed competitively and quickly in all datasets.

Comparison with previous studies: We compare our pipeline with published results on the same datasets and classifiers, as shown in table 6. Some published experiments used different parts of the datasets than what we used, thus we configured our experiments accordingly to compare fairly. For these comparisons, we used our best combination of pre-processing, feature extraction and selection methods and feature weighting. For SemEval, we used LibLinear instead of SVM and achieved marginally lower results than the published ones. Lastly, the method in [12] used 22,660 Senti-140 positive and negative instances. Since it is not mentioned which exaclty these instances were,

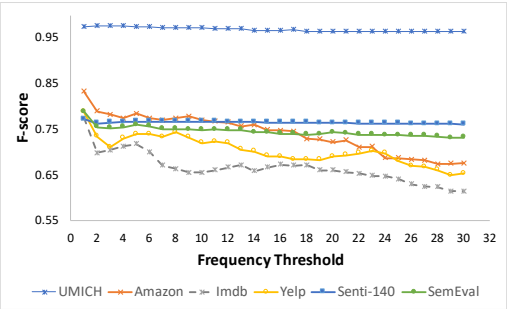

Fig. 1. F-score when using increasing frequency threshold values.

Table 6. Comparison between our sentiment analysis pipelines and state-of-the-art systems. Our scores have been computed using the same classifier, but different preprocessing and features (section 3). Abbreviations - TOK: tokenisation, Ngr: Ngrams, BoW: Bag-of-Words, SL: stoplist, PR: punctuation removal, U: unigrams, ST: stemming, LC: lowercasing, B: bigrams, SL: sentiment lexicon, LW: elongated words, NEG: negation.

		Method			Published scores (%)				Our scores (%)			
DataSet	Ref	Classifier	Preprocessing	Features	Acc	P	R	F1	Acc	P	R	F1
Amazon	[11]	NB	TOK	Tokens, Ngr	**82.4**	-	-	-	75.0	76.0	75.0	74.8
	[16]	NB	-	BoW	-	**78.9**	-	-	75.0	76.1	75.0	74.8
IMDB	[11]	NB	TOK	Tokens, Ngr	**78.6**	-	-	-	72.3	72.7	72.3	72.1
	[16]	NB	-	BoW	-	**78.9**	-	-	72.3	72.7	72.3	72.1
Yelp	[11]	NB	TOK	Tokens, Ngr	**82.7**	-	-	-	70.5	70.7	70.5	70.4
	[16]	NB	-	BoW	-	60.3	-	-	70.5	**70.7**	70.5	70.4
UMICH	[3]	SVM	TOK, SL, PR	U	-	-	-	89	99.1	99.1	99	**99.1**
SemEval	[15]	SVM	ST, LC	Ngr	**64.6**	-	**63.7**	**63.2**	62.9	60.7	59.3	59.9
Senti-140	[12]	SVM	BoW, clustering	U, B, SL, LW, NEG	-	-	-	77.4	80.5	80.0	80.0	**80.0**

we used the entire dataset with a frequency threshold of 100. We used the Liblinear classifier and the results were better by 2.6%.

Best performing model: CNB was the fastest classifier and often also performed best. It is beneficial for large datasets. The slowest classfier was RF. A combination of n-grams often performs best. The effect of frequency thresholding largely depends on the size of the data. Preprocessing matters and affects classification results. The best configuration, which achieved F-scores above 70% for all datasets, is the CNB model with tokeniser T1 and stemmer S1, all n-grams features and a frequency threshold of 6.

5 Conclusion

In this paper, we have investigated UIMA to optimise the accuracy and efficiency of sentiment analysis. We have demonstrated that UIMA can simplify the development of text-processing pipelines, wherein components can be freely combined using shared data types. We experimented with a wide range of pipeline configurations, considering various pre-processing components, classification algorithms, feature extraction methods and feature weighting schemes, to identify the best performing ones.

A potential limitation of our proposed sentiment analysis pipeline is that, like any other UIMA application, it is written as a sequential program, which limits its scalability. In the future we plan to leverage UIMA DUCC, i.e. the Distributed UIMA Cluster Computing platform, for scaling our sentiment analysis pipeline to big data collections. UIMA DUCC enables large-scale processing of big data collections by distributing a UIMA pipeline over a computer cluster while the constituent components of the pipeline can be executed in parallel across the different nodes of the cluster.

Acknowledgment

This research work is part of the TYPHON Project, which has received funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 780251.

References

1. Altrabsheh, N., Cocea, M., Fallahkhair, S.: Sentiment analysis: towards a tool for analysing real-time students feedback. In: ICTAI 2014. pp. 419–423. IEEE (2014)
2. Batista-Navarro, R., Carter, J., Ananiadou, S.: Argo: enabling the development of bespoke workflows and services for disease annotation. Database **2016** (2016)
3. Dridi, A., Recupero, D.R.: Leveraging semantics for sentiment polarity detection in social media. International Journal of Machine Learning and Cybernetics pp. 1–11 (2017)
4. Ferrucci, D., Lally, A.: UIMA: an architectural approach to unstructured information processing in the corporate research environment. Natural Language Engineering **10**(3-4), 327–348 (2004)
5. Go, A., Huang, L., Bhayani, R.: Twitter sentiment analysis. CS224N Project Report, Stanford (2009)
6. Greaves, F., Ramirez-Cano, D., Millett, C., et al.: Use of sentiment analysis for capturing patient experience from free-text comments posted online. Journal of medical Internet research **15**(11) (2013)
7. Khuc, V.N., Shivade, C., Ramnath, R., et al.: Towards building large-scale distributed systems for twitter sentiment analysis. In: Proceedings of SAC. pp. 459–464. ACM (2012)
8. Kontonatsios, G., Thompson, P., Batista-Navarro, R.T., et al.: Extending an interoperable platform to facilitate the creation of multilingual and multimodal nlp applications. In: Proceedings of ACL 2013: System Demonstrations. pp. 43–48 (2013)
9. Kotzias, D., Denil, M., De Freitas, N., et al.: From group to individual labels using deep features. In: Proceedings of ACM SIGKDD 2015. pp. 597–606. ACM (2015)
10. Mohammad, S.M., Kiritchenko, S., Zhu, X.: NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. arXiv preprint arXiv:1308.6242 (2013)
11. Pal, S., Ghosh, S.: Sentiment analysis using averaged histogram. International Journal of Computer Applications **162**(12) (2017)
12. Ren, Y., Wang, R., Ji, D.: A topic-enhanced word embedding for twitter sentiment classification. Information Sciences **369**, 188–198 (2016)
13. Rodrıguez-Penagos, C., Narbona, D.G., Sanabre, G.M., et al.: Sentiment analysis and visualization using UIMA and Solr. Unstructured Information Management Architecture (UIMA) p. 42 (2013)
14. Rosenthal, S., Farra, N., Nakov, P.: Semeval-2017 task 4: Sentiment analysis in twitter. In: Proceedings of SemEval-2017. pp. 502–518 (2017)
15. Sarker, A., Gonzalez, G.: Hlp @ upenn at semeval-2017 task 4a: A simple, self-optimizing text classification system combining dense and sparse vectors. In: Proceedings of SemEval-2017. pp. 640–643 (2017)
16. Sarma, P.K., Sethares, W.: Simple algorithms for sentiment analysis on sentiment rich, data poor domains. In: Proceedings of ACL 2018. pp. 3424–3435 (2018)
17. Sohn, S., Savova, G.K.: Mayo clinic smoking status classification system: extensions and improvements. In: AMIA Annual Symposium Proceedings. vol. 2009, p. 619. American Medical Informatics Association (2009)
18. UMICH: Dataset SI650 - sentiment classification (2011), `https://goo.gl/Xfr8lI`

19. Zavattaro, S.M., French, P.E., Mohanty, S.D.: A sentiment analysis of us local government tweets: The connection between tone and citizen involvement. Government Information Quarterly **32**(3), 333–341 (2015)